CENTRE DE RECHERCHE
SUR LES TRANSPORTS
(C.R.T.)
CENTRE FOR RESEARCH
ON TRANSPORTATION

UNIVERSITÉ DE MONTRÉAL
ÉCOLE DES HAUTES ÉTUDES COMMERCIALES
ÉCOLE POLYTECHNIQUE

CENTRE DE RECHERCHE SUR LES TRANSPORTS
UNIVERSITÉ DE MONTRÉAL
C.P. 6128, SUCCURSALE A,
MONTRÉAL
CANADA H3C 3J7

TÉLÉPHONE: (514) 343-7575
TÉLEX: 05-24146 BIBPOLYTEC
EMAIL: crt@crt.umontreal.ca
TÉLÉCOPIEUR (FAX): (514) 343-7121

CRT-679

A NEWTON-SOR METHOD FOR SOLVING THE SPATIAL PRICE EQUILIBRIUM PROBLEM

by

Patrice Marcotte[1]
Gérald Marquis[2]
Lourdes Zubieta[3]

[1] Centre de recherche sur les transports, Université de Montréal and Collège militaire royal de Saint-Jean.

[2] Centre de recherche sur les transports, Université de Montréal.

[3] Centre de recherche sur les transports, Université de Montréal and Bishop's University, Lennoxville, Quebec.

This research has been supported by the National Sciences and Engineering Research Council of Canada (grant A5789).

Centre de recherche sur les transports - Publication #679

January 1990

90-02984/

Dépôt légal - 1er trimestre 1990

ABSTRACT

In this paper we propose an efficient Newton-SOR algorithm for solving the separable spatial price equilibrium problem. Although an active-set strategy is implemented for solving for blocks of variables associated with origin nodes, global convergence of the iterates is not dependent on nondegeneracy or convexity assumptions. Numerical experiments indicate the excellent performance of the method on both degenerate and nondegenerate problems.

Key words : Networks, Equilibrium, Newton's method.

- - - - - - -

RÉSUMÉ

Nous proposons un algorithme de type Newton-SOR par blocs pour la résolution d'un problème d'équilibre spatial des prix sur un réseau bipartite. Chaque bloc de variables est traité à l'aide d'une stratégie de contraintes actives. La convergence globale de la méthode est indépendante d'hypothèses de convexité ou de non dégénérescence à une (la) solution. Les expériences numériques indiquent que notre algorithme se compare favorablement aux méthodes concurrentes.

Mots-clés : Réseaux, Équilibre, Méthode de Newton.

1. INTRODUCTION.

This paper describes an efficient method for solving a class of separable spatial price equilibrium models (SPEP). This general market equilibrium model has recently motivated a considerable amount of research since its early description by Samuelson (1952) and Takayama and Judge (1964, 1970). Special purpose algorithms have been primarily developed for cases where the supply and demand functions were linear. When the functions involved (supply, demand and transportation) are separable, the equilibrium conditions can be regarded as the Kuhn-Tucker optimality conditions of an optimization problem, whose critical points constitute market equilibria. Several solution approaches have been proposed in the literature and include fixed point or homotopy methods (MacKinnon, 1975), Benders decomposition (Polito, McCarl and Morin, 1980), nonlinear programming methods (Florian and Los, 1982; Zubieta, 1986), conjugate gradient algorithms (Pang and Lin, 1985; Drissi-Kaïtouni, 1987), relaxation methods (Güder, 1988; Nagurney, 1987; Eydeland and Nagurney, 1988; Crouzeix, Ferland and Zubieta, 1989), etc. When supply and demand functions are not integrable, in which case no simple equivalent optimization formulation exists, the problem has been solved as a complementarity problem (Mathiesen, 1974; Asmuth, Eaves and Peterson, 1979; Pang and Lee, 1981; Friesz, Tobin and Harker, 1983; Jones, Saigal and Schneider, 1984, 1985), or as a variational inequality problem (Fisk, 1987).

The algorithms presented in this paper are aimed at solving the class of market equilibrium problems where supply, demand and transportation cost functions are nonlinear and separable. Our approach consists in successive linearizations of the cost functions, each linearized subproblem being solved by a very fast block SOR (block Gauss-Seidel) scheme. The methods utilized can be viewed as an extension of the composite k-step Newton-SOR iteration commonly used in the solution of nonlinear equations (Ortega and Rheinboldt, 1970). The main feature of market equilibrium problems is that the variables are restricted solely by nonnegativity constraints and that advantage can be drawn from the bipartite network structure of the problem.

The idea of using block SOR iterations in this context is not new. Actually it has been independently proposed by Nagurney (1987) and Zubieta (1986) for solving linear and nonlinear SPEPs, respectively. Nagurney in particular has successfully used this approach in her adaptation of the Dafermos and Sparrow (1969) equilibration scheme and, more recently, has specialized the algorithm for the linear case (Dafermos and Nagurney, 1988). On the other hand, Zubieta (1986) has proposed a block SOR-Newton approach that incorporates an active set strategy for solving separable nonlinear SPEPs. In this paper, the choice of an efficient algorithm has been guided by the following key considerations:

(i) Sparse, separable SPEPs are best solved by a combination of SOR and active set

strategy.

(ii) Linear SPEPs can be solved much faster than nonlinear ones, thus suggesting iteratively linearizing the nonlinear problem.

The use of active set strategies usually requires some sort of nondegeneracy or strict complementarity assumption. However, transposing a result obtained by Marcotte (1984) in the context of oligopoly models we could relax this assumption.

The paper is organized as follows: Mathematical formulations of the problem are introduced in section 2; in section 3 we present a basic version of our algorithm; section 4 addresses the issues of global and local convergence; section 5 discusses the implementation of the basic algorithm actually used; section 6 discusses the computational complexity of the algorithm; finally, numerical results are presented in section 7.

2. THE NONLINEAR MODEL.

In this section we review the single commodity, nonlinear SPEP with distinct supply and demand price functions. We first introduce some notations and assumptions.

Y_i:	total supply (output) at origin (market) $i \in I = \{1, \ldots, m\}$
Y:	supply vector $\{Y_i\}_{i \in I}$
Z_j:	total demand (input) at destination (market) $j \in J = \{1, \ldots, n\}$
x_{ij}:	quantity shipped (traded) between origin $i \in I$ and destination $j \in J$
Z:	demand vector $\{Z_j\}_{j \in J}$
$S_i(Y_i)$:	supply function at origin i
$I_j(Z_j)$:	demand function at destination j
$c_{ij}(x_{ij})$:	transportation (transaction) cost between origin i and destination j

Assumption A1: The supply, demand and transportation cost functions are nonnegative and once continuously differentiable over the nonnegative real axis. Moreover, supply and transportation are nondecreasing, while demand is nonincreasing.

Assumption A2: $\lim_{Y_i \to \infty} S_i(Y_i) = 0$ and $\lim_{Z_j \to \infty} I_j(Z_j) = +\infty$.

Assumption A1 is standard in our context. Assumption A2, a coercivity condition, ensures (i) the existence, via the Kakutani fixed point theorem, of at least one solution to the problem (ii) the boundedness of the solution set. In particular there are no solutions "at infinity".

Assuming perfect competition between the markets, an equilibrium state will be achieved when, for all pairs of supply and demand markets, the demand price equals

the sum of supply and transportation prices, if trade takes part between the two markets, and will not be less than the demand price if trade does not occur. Mathematically the equilibrium can be characterized as a triple $\{(Y_i)_{i\in I}, (Z_j)_{j\in J}, (x_{ij})_{i\in I, j\in J}\}$ that solves the system of equalities and inequalities:

$$S_i(Y_i) + c_{ij}(x_{ij}) \begin{cases} = I_j(Z_j), & \text{if } x_{ij} > 0; \\ \geq I_j(Z_j), & \text{if } x_{ij} = 0. \end{cases}$$

$$Y_i = \sum_{j\in J} x_{ij}, \qquad \forall i \in I$$

$$Z_j = \sum_{i\in I} x_{ij}, \qquad \forall j \in J$$

$$x_{ij} \geq 0, \qquad \forall i, j \in I \times J.$$

If assumptions A1 and A2 hold, the equilibrium conditions (1) are the Karush-Kuhn-Tucker optimality conditions for the minimization problem

$$\min_{Z\geq 0} G(Z) = \sum_{i\in I} \int_0^{\sum_{j\in J} x_{ij}} S(t)\,dt + \sum_{i\in I}\sum_{j\in J} \int_0^{x_{ij}} c_{ij}(t)\,dt - \sum_{j\in J} \int_0^{\sum_{j\in J} x_{ij}} I(t)\,dt \qquad (3)$$

where $X = (x_{ij})_{i\in I, j\in J}$, or, equivalently, as

$$\min \sum_{i\in I} \int_0^{Y_i} S(t)\,dt + \sum_{i\in I}\sum_{j\in J} \int_0^{x_{ij}} c_{ij}(t)\,dt - \sum_{j\in J} \int_0^{Z_j} I(t)\,dt \qquad (4)$$

where Y_i and Z_j satisfy the condition (2) above. Existence of at least one solution to (3) or (4) is guaranteed by assumption A2 which rules out the possibility of a minimum at infinity (horizontal asymptote).

3. THE BASIC ITERATIVE FRAMEWORK.

The solution of the nonlinear complementarity problem (1) or its equivalent minimization problem (2) will be achieved by solving iteratively a sequence of linearized problems, each of which will be solved by a block-SOR algorithm. We thus obtain algorithm N

ALGORITHM N ("N" for "Newton")

Let ϵ be a predetermined tolerance.

Let x_0 denote an initial feasible solution.

Set $k = 0$.

1. ITERATION k

Solve a linearized SPEP with supply, demand and transportation cost functions given by:

$$\begin{aligned} \overline{S}_i(Y_i) &\stackrel{\text{def}}{=} S_i(Y_i^k) + S_i'(Y_i^k)(Y_i - Y_i^k) \\ \overline{c}_{ij}(x_{ij}) &\stackrel{\text{def}}{=} c_{ij}(x_{ij}^k) + c_{ij}'(x_{ij}^k)(x_{ij} - x_{ij}^k) \\ \overline{I}_j(Z_j) &\stackrel{\text{def}}{=} I_j(Z_j^k) + I_j'(Z_j^k)(Z_j - Z_j^k) \end{aligned} \tag{5}$$

i.e. find x, Y, Z such that

$$\overline{S}_i(Y_i) + \overline{c}_{ij}(x_{ij}) \begin{cases} = \overline{I}_j(Z_j), & \text{if } x_{ij} > 0 \\ \geq \overline{I}_j(Z_j), & \text{if } x_{ij} = 0 \end{cases}$$

$$\begin{aligned} Y_i &= \sum_{j \in J} x_{ij} \qquad \forall i \in I \\ Z_j &= \sum_{i \in I} x_{ij} \qquad \forall j \in J \\ x_{ij} &\geq 0 \qquad \forall i, j \in I \times J. \end{aligned} \tag{6}$$

Let $\overline{\epsilon}(x) \stackrel{\text{def}}{=} \sum_{i \in I} \sum_{j \in J} |S_i(Y_i^k) + c_{ij}(x_{ij}) - I_j(Z_j)|$.

If $\overline{\epsilon}(x) < \epsilon$

then STOP

else update solution:

$$x^{k+1} \leftarrow x; Y^{k+1} \leftarrow Y; Z^{k+1} \leftarrow Z; k \leftarrow k+1$$

and return to 1.

□

Each linearized problem is solved inexactly using a block-SOR scheme, the ith block of variables corresponding to the vector of variables $X_i \stackrel{\text{def}}{=} (x_{ij})_{j \in J}$ associated with each origin or supply market $i \in I$. The resulting algorithm is outlined below.

ALGORITHM *BSOR*

Let ϵ' be a predetermined tolerance.

Let X^0, Y^0 and Z^0 be the corresponding initial solution vector.

Set $l = 0$.

1. ITERATION l (solution of linearized block l):

Solve:

$$\min_{X_l} \overline{G}(X_1^l, \ldots, X_{l-1}^l, X_l, X_{l+1}^{l-1}, \ldots, X_m^{l-1}). \tag{7}$$

if $l < m$

then set $l = l + 1$ and go to 1

else

$$X_i^0 \leftarrow X_i^m, i \in I$$
$$Y_i^0 \leftarrow Y_i^m, i \in I$$
$$Z_j{}^0 \leftarrow Z_j{}^m, j \in J.$$

let $\epsilon_{ij}(x) = \begin{cases} |S_i(Y_i^0) + c_{ij}{}^0(x_{ij}) - I_j(Z_j{}^0)| & \text{if } x_{ij} > 0 \\ -(S_i(Y_i^0) + c_{ij}(x_{ij}{}^0) - I_j(Z_j{}^0) & \text{if } x_{ij} = 0 \end{cases}$

and $\epsilon'(x) = \sum_{i \in I} \sum_{j \in J} \epsilon_{ij}(x_{ij})$.

if $\epsilon'(x) < \epsilon'$

then set $Y_i = Y_i{}^m, i \in I$ and STOP

else set $l = 0$ and go to 1.

□

Remarks.

1. If $n < m$ it will in general be more efficient to solve the problem by destinations, resulting in an obvious, symmetrical method.

2. The convergence criterion is a very stringent one and has been adopted to be consistant with Zubieta (1987) and Dafermos and Nagurney (1989). A more manageable criterion could have been based on the global quantity:

$$\sum_{i \in I} \sum_{j \in J} x_{ij} |S_i(Y_i^0) + c_{ij}(x_{ij}{}^0) - I_j(Z_j{}^0)|.$$

To complete the description of the overall method, there remains to specify how the minimization problem (7) is solved. As observed by Zubieta (1987), this problem can be solved in closed form, provided that the set of active (nonzero) variables is known a priori. Assume this is the case and denote by J_l the subset of J corresponding to the active variables x_{lj}. Figure 1 illustrates a block equilibrium for block 3 where $J_3 = \{2, 3, 4\}$. In these circumstances, (7) is equivalent to solving the set of linear equations (we drop the superscripts to enhance understanding):

$$\overline{S}_l(Y_l) = \overline{I}_j(Zj) - \overline{c}_{lj}(x_{lj}) \qquad \forall j \in J_l$$

or

$$\overline{S}_l(\sum_{j \in J_l} x_{lj}) = \overline{I}_j(x_{lj} + \sum_{i \neq l} x_{ij}) - \overline{c}_{lj}(x_{lj}) \quad \forall j \in J_l.$$

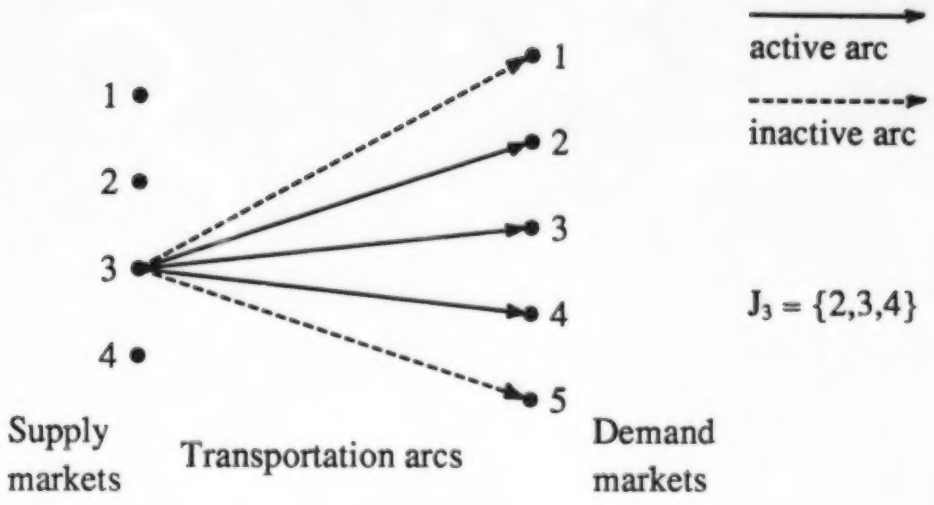

Figure 1. Block equilibrium (linearized costs).

We rewrite this system of equations as

$$\alpha_l + \beta_l Y_l = a_j - b_j x_{lj} \tag{8a}$$

$$Y_l = \sum_{j \in J} x_{lj} \tag{8b}$$

where

$$\alpha_l = S_l(Y_l^k) - S_l'(Y_l^k) Y_l^k$$

$$\beta_l = S_l'(Y_l^k)$$

$$a_j = c_{lj}(x_{lj}^k) - c_{lj}'(x_{lj}^k) x_{lj}^k$$

$$b_j = c_{lj}'(x_{lj}^k).$$

Solving for x_{lj} in (8a) there comes easily:

$$x_{lj} = \frac{a_j - \alpha_l - \beta_l Y_l}{b_j}.$$

Substituting in (8b) we obtain:

$$Y_l = \sum_{j \in J} \frac{a_j - \alpha_l}{b_j} - Y_l \sum_{j \in J} \frac{\beta_l}{b_j},$$

hence

$$Y_l = \frac{\sum_{j\in J}(a_j - \alpha_l)/b_j}{1 + \sum_{j\in J}\beta_l/b_j}$$

and finally

$$x_{lj} = \frac{a_j - \alpha_l}{b_j} - \frac{\beta_l}{b_j}\frac{\sum_{j\in J}(a_j - \alpha_l/)b_j}{1 + \sum_{j\in J}\beta_l/b_j}.$$

The difficulty consists in determining, for each origin (block) l, the right set of active constraints indices J_l. A standard way of dealing with this problem would be to use an active set strategy, i.e. to modify the set J_l by inserting or deleting one variable at a time, according to specific rules (see Luenberger, 1986). However the standard implementation of this strategy usually requires a nondegeneracy assumption, the so-called strict complementarity hypothesis:

$$x^*_{ij}[S_i(Y^*_I) + c_{ij}(x^*_{ij}) - I_j(Z^*_J)] = 0$$

at a solution (x^*, Y^*, Z^*). However, as we will see in the numerical examples, degeneracy occurs, especially when the transportation cost functions $c_{ij}(x_{ij})$ are constant, a most common situation. To dispense with this assumption we propose an algorithm which, for a linearized problem, converges in a finite number of iterations.

ALGORITHM *AS* **(Active set Strategy) FOR BLOCK (ORIGIN)** i

Set $J_i = J$.

1.

$$\text{Let } x_{lj} = \frac{a_j - \alpha_l}{b_j} - \frac{\beta_l}{b_j}\frac{\sum\limits_{j\in J}(a_j - \alpha_l)/b_j}{1 + \sum\limits_{j\in J}\beta_l/b_j}, \qquad \forall j \in J_l.$$

2. Let $\overline{J} = \{j | x_{lj} \leq 0\}$.

3. **if** $\overline{J} = \emptyset$
   then STOP (optimal solution found for linearized block l)
   else $J_l \leftarrow J_l - \overline{J}$ and return to 1. □

The preceding algorithm is inspired from an algorithm developed in the context of oligopolistic market equilibrium (see Marcotte, 1988). The proof of exactness of the method is based on a monotonicity principle, and is given below.

Proposition 1. When algorithm *AS* stops, $X_l = (x_{lj})_{j\in J}$ is solution to (7), i.e.

$$\overline{S}_l(Y_l) + \overline{c}_{lj}(x_{lj}) = \overline{I}_j(Z_j) \qquad \forall j \in J_l \tag{9a}$$

$$\geq \overline{I}_j(Z_j) \qquad \forall j \in J - J_l. \tag{9b}$$

Proof. For ease of exposition, let us temporarily drop the origin index l. We want to solve the linear system:

$$\begin{aligned} \alpha + \beta Y &= a_j - b_j x_j \\ Y &= \sum_{j \in J^*} x_j \\ x_j &\geq 0 \end{aligned} \tag{10}$$

where J^* is the maximal index set having the property that

$$j \in J - J^* \Rightarrow a_j - b_j x_j - \beta Y \leq 0.$$

For a fixed index set J, the solution to (10) is given by

$$x_j = \frac{a_j - \alpha - \beta Y}{b_j} \tag{11}$$

$$Y = \frac{\sum_{j \in J} (a_j - \alpha)/b_j}{1 + \beta \sum_{j \in J} 1/b_j} \tag{12}$$

Without loss of generality one may assume that Y is nonnegative. Indeed, if this were not the case, we would have that the quantity $a_j - \alpha - \beta Y$ is strictly negative for $j \in \overline{J}$. We could deduce that $a_j - \alpha$ is strictly negative for $j \in \overline{J}$ and conclude that every index $j \in \overline{J}$ is not a member of the optimal index set J^* and could consequently be deleted.

In the following, let $\overline{J}$ stand for the index set corresponding to the variables deleted at the very first iteration and let

$$\overline{x}_j = \frac{a_j - \alpha - \beta \overline{Y}}{b_j} \tag{13}$$

$$\overline{Y} = \frac{\sum_{j \in \overline{J}} (a_j - \alpha)/b_j}{1 + \beta \sum_{j \in \overline{J}} 1/b_j}. \tag{14}$$

The proof will be complete if we can show that $\overline{Y} \geq Y$. Indeed, if this is the case, the algorithm will produce a sequence of increasing Y-values. Denote by Y^* the last Y-value obtained and let x_j be a variable that was deleted at a stage corresponding to a Y-value $\tilde{Y} \leq Y^*$. We have:

$$\frac{a_j - \alpha - \beta Y^*}{b_j} \leq \frac{a_j - \alpha - \tilde{Y}}{b_j}$$

and we conclude that (10) is satisfied for Y^*. In other words, no deleted variable can be part of the optimal solution at a positive level. The proof that $\overline{Y} \geq Y$ is technical and left for the appendix. □

One iteration of the algorithm for minimizing $G(X_1, \ldots, X_m)$ now takes the form:

ALGORITHM *NBSOR* (Newton-Block-SOR)

1. Linearize $G(x)$ at the current iterate x^k.
2. Solve the linearized problem $\min_x \overline{G}(x)$ to obtain $\overline{x}$ using algorithm *BSOR*.
3. Solve each block using algorithm *AS*.
4. Perform linesearch:

$$x^{k+1} \in \arg \min_{x \in [x^k, \overline{x}]} G(x).$$

□

At the fourth step, any suitable linesearch, such as the Armijo-Goldstein rule, can be implemented.

5. CONVERGENCE ANALYSIS.

Since the cost functions involved are all continuously differentiable, global convergence will be guaranteed by the closedness of the direction-finding and linesearch procedures used (see Luenberger, 1986). Global convergence of the proposed method does not hinge on convexity of the function G, it may very well be that the equilibrium achieved is not unique (local minimum of G) or even unstable, when the Hessian matrix of the cost function is indefinite at the solution (stationary point).

The rate of convergence of Newton's method is quadratic, provided that strict complementarity is satisfied and that the Hessian of G is invertible in a neighborhood of the solution. However, in our context, this result has only theoretical value since the linearized problems are not solved exactly. If L iterations of the Block-SOR algorithm are performed, then we can quote the following result for the resulting L-step-NBSOR method.

Proposition. If strict complementarity is satisfied and there exist a solution x^* and a neighborhood $V_\epsilon(x^*)$ such that G is strongly convex in $V(x^*)$, then there exists a neighborhood $\overline{V}(x^*)$ of x^* such that x^k converges to x^* (at least) linearly, provided that $x^k \in \overline{V}(x^*)$ for some index K.

Proof. From the strict complementarity and strong convexity assumptions we deduce the existence of a neighborhood $\tilde{V}(x^*)$ such that

$$x^k \in \tilde{V}(x^*) \quad \forall k \geq K$$
$$x^k_{ij} = 0 \iff x^*_{ij} = 0 \quad \forall k \geq K.$$

Set $\overline{V}(x^*) = \hat{V}(x^*)$. For $k \geq K$, $x^* \in \overline{V}(x^*)$, the active constraints are identified, and the algorithm L-step-NBSOR merely solves the system of nonlinear equations:

$$S_i(\sum_{j \in J} x_{ij}) + c_{ij}(x_{ij}) - I_j(\sum_{i \in I} x_{ij}) = 0 \quad \text{if} \quad {x_{ij}}^* > 0$$

$$x_{ij} = 0 \quad \text{if} \quad {x_{ij}}^* = 0.$$

Local convergence at a (at least) linear rate then follows from standard results in numerical analysis (see Ortega and Rheinboldt, 1970, page 321). □

6. IMPLEMENTATION.

The previously proposed algorithmic framework leaves much freedom for algorithmic design. For example, in our implementation, the number of iterations on the linearized subproblems have been increased at each global iteration according to the formula:

$$l_k = 1 + k,$$

the rationale being that, close to an equilibrium solution, the linearized problem constitutes a good approximation to the initial nonlinear problem, and that the linearizing process is costly (see next section). Although it is possible to demonstrate rigorously that increasing L_k may lead better-than-linear convergence for the algorithm (see Ortega and Rheinboldt, 1970), we only used this scheme in a heuristic fashion.

Another key modification brought to the general algorithm is that only the first iteration of the Block-SOR scheme is performed completely, i.e. including all variables. The $L_k - 1$ remaining iterations only involve the variables that assumed positive values at the end of the first cycle. This active-set strategy induces a great reduction in the amount of work done by algorithm AS, especially for problems where the percentage of nonzero equilibrium variables is low, which is typical of many real-life problems. The first iteration, however, must include all flow variables in order to preserve global convergence of the overall algorithm. A proof of convergence of this modified algorithm follows from a straightforward application of the so-called "spacer step theorem" (see Luenberger, 1986, p. 231).

7. COMPUTATIONAL COMPLEXITY.

The reader might well ask her or himself why a Newton-SOR scheme (outer linearization) is preferable to SOR-Newton (inner linearization) as proposed by several researchers. Indeed it would seem more efficient to use the latest nonlinear information available before linearizing, rather than iterating several times over an approximate (linearized) model.

Some insight can be gained by comparing the number of iterations required by each approach and multiplying by the amount of work required at each iteration. To be specific, we will compare the amount of work required by

(i) one outer iteration of our algorithm

(ii) one cycle of the L-step SOR-Newton algorithm, where each nonlinear block is solved using Newton's method.

We will in the sequel denote by ρ_1 the amount of work (CPU time) required by the evaluation of one of the functions S_i, I_j, c_{ij} or one of its derivatives, and by ρ_2 amount of work required to perform one elementary operation: addition, subtraction, multiplication or division. The analysis is summarized in Tables 1 and 2.

LINEARIZATION:

$$S_i: \quad 2m(\rho 1 + \rho 2)$$

$$c_{ij}: \quad 2mn(\rho_1 + \rho_2)$$

$$I_j: \quad 2n(\rho_1 + \rho_2)$$

FIRST COMPLETE SOR SUBITERATION:

- computation of $Y_l = \sum_{j \in J}(a_j - \alpha_l/b_j)/(1 + \sum_{j \in J}\frac{\beta_l}{b_j})$: $5n\rho_2$

- computation of $x_{lj} = a_j - \alpha_l - \beta_l Y_l/b_j$: $4n\rho_2$.

REMAINING $L-1$ INCOMPLETE SOR SUBITERATIONS: $9m\nu_k\rho_2$.

TOTAL: N_1=$K[(11mn + 9Lm\nu_k)\rho_2 + 2mn\rho_1]$

Table 1. Time complexity of one $NBSOR$ iteration.

If both methods use the same number of outer and inner iterations respectively, i.e. $L_1 = l$ and $K_1 = K$ and we set $\nu_k = n$ (no active set strategy) and $\rho_1 \approx \rho_2$ then we obtain, only keeping the high-order terms:

$$N_1 \approx 9KLmn$$

$$N_2 \approx 17KLmn.$$

LINEARIZATION OF ONE BLOCK:

LINEARIZATION OF S_i: $2m(\rho_1 + \rho_2)$

LINEARIZATION OF c_{ij}: $2mn(\rho_1 + \rho_2)$

LINEARIZATION OF I_j: $2mn(\rho_1 + \rho_2)$

RESOLUTION OF EACH LINEARIZED BLOCK: $9mn\rho_2$

TOTAL: $K_1 L_1[4mn(\rho_1 + \rho_2) + 9mn\rho_2]$

$N_2 = 4mnK_1L_1[4\rho_1 + 13\rho_2]$.

Table 2. Time complexity of one SOR-Newton iteration.

Under those assumptions on K, L, K_1, L_1, ρ_1, ρ_2 and ν_k, algorithm Newton-SOR is seen to require slightly less work than algorithm SOR-Newton. One might think however that this advantage is offset by the fact that SOR-Newton might be expected to converge slightly faster (in less iterations) to the optimal solution. In most practical situations however, we will have $\rho_1 \leq \rho_2$ (evaluation is more costly than an elementary operation) and $\nu_k \leq m$, thus providing a theoretical explanation for the better behaviour of Newton-SOR observed in the preceding section.

7. NUMERICAL RESULTS.

Several numerical experimentations have been performed on pseudo-randomly generated problems where the supply and demand functions assumed the respective mathematical forms:

$$S_i(Y_i) = u_i + (S_i/v_i)^2$$

$$I_j(Z_j) = \frac{1}{\theta_j} \ln\left(\frac{2000}{Z_j}\right)$$

where u_i, v_i and θ_j are pseudo-random uniform variates on the intervals $[3,5]$, $[15,20]$ and $[.1,.5]$ respectively. The algorithms have been coded in Pascal on a SUN 3/50 microcomputer running under the UNIX operating system.

The first set of problems is identical with those reported in Zubieta (1986). For this set of problems we compared our algorithm with a SOR-Newton method as well as an active set strategy (GSNA) variant of Zubieta's algorithm whose behaviour proved quite similar. The transportation cost functions assumed the linear form:

$$c_{ij}(x_{ij}) = \gamma_{ij} x_{ij}$$

with coefficients γ_{ij} uniformly distributed on the interval $[5,10]$. The convergence parameter ϵ was set to 10^{-5}.

Since unit arc costs are constant for this set of problems, the subset of basic link flows forms a spanning tree. Degeneracy occurs when the set of arcs with nonzero flow is less than $m + n - 1$, one less than the total number of markets. This phenomenon was observed in some of the test problems but did not, as could have been the case with the stopping criterion chosen, prevent the algorithm to function correctly. Numerical results are reported in tables 3 and 4.

The second set of test problems was similarly designed but involved a larger number of markets (see tables 5,6,7,8). In comparison, Dafermos and Nagurney (1988) report CPU times of the order of five seconds for solving 50×50 purely linear problems on a CYBER 870 mainframe computer, with the accuracy parameter ϵ set at 1. Our times (roughly 50 CPU seconds for an $\epsilon = .01$ accuracy) on nonlinear problems compare favorably, especially when one remembers that a large share of computing time is spent in linearizing the cost functions.

Finally a third set of problems involving nonlinear, convex arc cost of the form:

$$c_{ij}(x_{ij}) = (x_{ij}/c_{ij})^4$$

was solved. As expected, running times increased sharply, due mainly to two factors:

(i) The increased density of the solution; indeed, most arcs carried flow at equilibrium.

(ii) The nonlinearity of the arc cost functions, which made accurate solutions much more difficult to obtain. For the largest problem, the convergence criterion had to be weakened ($\epsilon = .01$) to reduce computer times.

In spite of those numerical difficulties, problems of significant sizes could be solved in short amount of time on the SUN microcomputer. Numerical results for the entirely nonlinear test problems are reported in Table 9.

8. CONCLUSION.

In this paper we have presented an efficient implementation of a Newton-SOR method for solving the nonlinear SPEP. We believe this implementation to be quasi-optimal for a class of problems with a relatively large percentage of nonzero equilibrium flows. In extreme cases (very few or very many nonzero equilibrium flows) our method would probably be outperformed, as far as local convergence goes, by a true Newton-active set method (see Drissi-Kaïtouni, 1988).

We think that, due to its inherent simplicity and robustness, this algorithm would be ideally suited as a building block in designing efficient algorithms for solving nonseparable SPEPs in the framework of a global Jacobi or Gauss-Seidel scheme for instance. This will be discussed in future experimentations.

REFERENCES

ASMUTH, B., EAVES. B.C., PETERSON, E.L. (1979) "Computing economic equilibria on affine networks", *Mathematics of Operations Research* **4**, 209–214.

CROUZEIX, J.-P., FERLAND, J.A., ZUBIETA, L. (1989) "Gauss-Seidel- Newton-Armijo approach for minimization problems on the nonnegative orthant. Application to Spatial Price Equilibrium Problems", Publication , Centre de recherche sur les transports, Université de Montréal (forthcoming).

DAFERMOS, S., SPARROW, F.T. (1969) "The traffic assignment problem for a general network", *Journal of Research of the National Bureau of Standards* **73B**, 2, 91–118.

DAFERMOS, S., NAGURNEY, A. (1989) "Supply and demand equilibration algorithms for a class of market equilibrium problems", *Transportation Science* **23**, 2, 118–124.

DRISSI-KAÏTOUNI, O. (1988) "Applications de méthodes de Newton au problème d'équilibre spatial sur un réseau général", Ph.D. thesis, Département d'Informatique et de Recherche Opérationnelle, Université de Montréal.

FISK, C.S. (1987) "Spatial Price Equilibrium on congested networks", *Transportation Research B* **21B**, 3, 175–182.

FLORIAN, M., LOS, M. (1982) "A new look at static Spatial Price Equilibrium models", *Regional Science and Urban Economics* **12**, 579–597.

GÜDER, F. (1989) "Pairwise reactive SOR algorithms for quadratic programming of Net Import Spatial Equilibrium models", *Mathematical Programming* **14**, 175–186.

JONES, P.C., SAIGAL, R., SCHNEIDER, M.H. (1984) "A variable dimension homotopy for computing spatial equilibria", *Operations Research Letters* **3**, 19–24.

JONES, P.C., SAIGAL, R., SCHNEIDER, M.H. (1985) "Computing nonlinear network equilibria", *Mathematical Programming* **31**, 57–66.

LUENBERGER, D.G. (1984) Linear and nonlinear programming, second edition, Addison-Wesley.

MACKINNON, J.G. (1975) "An algorithm for the generalized transportation problem", *Regional Science and Urban Economics* **5**, 445–464.

MARCOTTE, P. (1984) "Quelques notes et résultats nouveaux sur le problème d'équilibre d'un oligopole", *R.A.I.R.O. Recherche Opérationnelle* **18**, 2, 147–171.

MATHIESEN, L. (1974) "A linear complementarity approach to general equilibrium", report WP-SOL-74-9, Dept. of Operations Research, Stanford University.

NAGURNEY, A. (1987) "Computational comparisons of Spatial Price Equilibrium methods", *Journal of Regional Science* **27**, 55–76.

ORTEGA, J.M., RHEINBOLDT, W.C. (1970) "Iterative solution of nonlinear equations in several variables", Academic Press.

PANG, J.S., LEE, P.L. (1981) "A parametric linear complementarity technique for the computation of equilibrium prices in single commodity trade flows", *Mathematical Programming* **20**, 98–107.

PANG, J.S., LIN, Y.Y. (1985) "A dual conjugate gradient method for the single commodity Spatial Equilibrium Problem", *Spatial Price Equilibrium: Advances in theory, computation and applications*, edited by P.T. Harker, Springer Verlag, 136–157.

POLITO, J., McCARL, B.A., MORIN, T.L. (1980) "Solution of Spatial Equilibrium Problems with Benders decomposition", *Management Science* **26**, 593–605.

SAMUELSON (1952) "Spatial price equilibrium and linear programming", *American Economic Review* **42**, 283–303.

TAKAYAMA, T., JUDGE, G.G. (1964) "Equilibrium among spatially separated markets: A reformulation", *Econometrica* **32**, 510–524.

TAKAYAMA, T. JUDGE, G.G. (1970) "Alternative Spatial Equilibrium Models", *Journal of Regional Science* **10**, 1–12.

ZUBIETA, L. (1986) "Modèles d'équilibre spatial et méthodes de résolution", Ph.D. thesis, département d'Informatique et de Recherche Opérationnelle, Université de Montréal.

APPENDIX

Proof that $\overline{Y} \geq Y$.

$$
\begin{aligned}
Y &= \left(\sum_{j\in J}\frac{a_j-\alpha}{b_j}\right)\Big/\left(1+\beta\sum_{j\in J}\frac{1}{b_j}\right) \\
&= \left(\sum_{j\in \overline{J}}\frac{a_j-\alpha}{b_j}+\sum_{j\in J-\overline{J}}\frac{a_j-\alpha}{b_j}\right)\Big/\left(1+\sum_{j\in \overline{J}}\frac{\beta}{b_j}+\sum_{j\in J-\overline{J}}\frac{\beta}{b_j}\right) \\
&= \left(\sum_{j\in \overline{J}}\frac{a_j-\alpha}{b_j}+\sum_{j\in J-\overline{J}}\frac{\overline{Y}}{b_j}+\sum_{j\in J-\overline{J}}\frac{a_j-\alpha-\overline{Y}}{b_j}\right)\Big/\left(1+\sum_{j\in \overline{J}}\frac{\beta}{b_j}+\sum_{j\in J-\overline{J}}\frac{\beta}{b_j}\right).
\end{aligned}
$$

Setting $\delta = \sum_{j\in J-\overline{J}}(a_j-\alpha-\overline{Y})/b_j \leq 0$ and replacing $\overline{Y}$ by its expression there comes:

$$
\begin{aligned}
Y &= \frac{\sum_{j\in \overline{J}}\frac{a_j-\alpha}{b_j}+(\sum_{j\in J-\overline{J}}\frac{\beta}{b_j})(\sum_{j\in \overline{J}}\frac{a_j-\alpha}{b_j})/(1+\sum_{j\in J-\overline{J}}\frac{\beta}{b_j})}{1+\sum_{j\in J-\overline{J}}\frac{\beta}{b_j}+\sum_{j\in J-\overline{J}}\frac{\beta}{b_j}} \\
&= \frac{\sum_{j\in \overline{J}}\frac{a_j-\alpha}{b_j}}{1+\sum_{j\in J-\overline{J}}\frac{\beta}{b_j}}\left[\frac{1+(\sum_{j\in J-\overline{J}}\frac{\beta}{b_j})/(1+\sum_{j\in J-\overline{J}}\frac{\beta}{b_j})+\delta/(\sum_{j\in \overline{J}}\frac{a_j-\alpha}{b_j})}{1+(\sum_{j\in J-\overline{J}}\frac{\beta}{b_j})/(1+\sum_{j\in J-\overline{J}}\frac{\beta}{b_j})}\right] \\
&= \overline{Y}\left[1+\frac{\delta}{(\sum_{j\in \overline{J}}\frac{a_j-\alpha}{b_j})(1+(\sum_{j\in J-\overline{J}}\frac{\beta}{b_j})/(1+\sum_{j\in J-\overline{J}}\frac{\beta}{b_j}))}\right].
\end{aligned}
$$

Now we have that

$$\delta \leq 0, \tag{A1}$$

$$1+\frac{\sum_{j\in J-\overline{J}}\frac{\beta}{b_j}}{1+\sum_{j\in \overline{J}}\frac{\beta}{b_j}} \geq 0, \tag{A2}$$

$$Y \geq 0 \Rightarrow \sum_{j\in J}\frac{a_j-\alpha}{b_j} \geq 0 \Rightarrow \sum_{j\in \overline{J}}\frac{a_j-\alpha}{b_j} \geq 0, \tag{A3}$$

since all variables for which the quantity $a_j-\alpha-\beta Y$ is negative were deleted. From (A1), (A2) and (A3) we conclude that $\overline{Y} > Y$. □

PROB	SOR-Newton		GSNA		*NBSOR*	
	#iterations	CPU Time(sec)	#iterations	CPU Time(sec)	#iterations	CPU Time(sec)
5.1	60	6.2	61	3.7	11	1.5
5.2	36	3.8	30	1.9	5	0.9
5.3	54	5.7	39	2.2	11	1.3
5.4	59	5.9	108	5.6	14	1.9
5.5	33	3.4	30	1.8	8	0.8
10.1	33	12.3	41	6.7	10	3.1
10.2	34	12.3	74	10.8	8	2.2
10.3	77	27.6	129	17.0	13	4.2
10.4	39	14.5	47	7.4	9	2.5
10.5	162	58.3	80	11.8	11	3.7
15.1	222	167.3	224	51.6	21	15.8
15.2	30	23.7	44	15.5	11	7.5
15.3	91	69.7	104	26.7	14	9.2
15.4	560	417.6	205	49.0	45	52.1
15.5	49	38.0	78	22.1	10	6.6
20.1	223	288.2	264	86.9	21	23.1
20.2	1031	1317.8	235	82.5	26	30.9
20.3	99	128.3	110	48.3	17	16.4
20.4	108	139.9	187	69.2	21	21.8
20.5	126	163.6	170	65.2	18	17.8

Table 3. Small size problems.

NBSOR				
PROB	$x^0 = 4e$ # iterations (CPU Time in sec)	$x^0 = 10e$ # iterations (CPU Time in sec)	$x^0 = 16e$ # iterations (CPU Time in sec)	$x^0 = 30e$ # iterations (CPU Time in sec)
20.1	22(24.3)	21(23.1)	20(20.8)	23(25.2)
20.2	21(23.0)	26(30.9)	33(43.3)	26(30.0)
20.3	20(20.1)	17(16.4)	16(15.0)	19(17.4)
20.4	19(19.3)	21(21.8)	15(14.1)	21(21.1)
20.5	21(21.1)	18(17.8)	14(13.2)	17(15.8)
GSNA (Active Constraints)				
PROB	$x^0 = 4e$ # iterations (CPU Time in sec)	$x^0 = 10e$ # iterations (CPU Time in sec)	$x^0 = 16e$ # iterations (CPU Time in sec)	$x^0 = 30e$ # iterations (CPU Time in sec)
20.1	228(82.7)	264(86.9)	260(90.6)	264(94.1)
20.2	735(229.9)	235(82.5)	224(80.4)	570(182.0)
20.3	108(46.1)	110(48.3)	535(164.3)	677(207.2)
20.4	190(69.7)	187(69.2)	198(73.3)	209(74.8)
20.5	161(62.9)	170(65.2)	178(68.2)	174(67.4)
SOR-Newton				
PROB	$x^0 = 4e$ # iterations (CPU Time in sec)	$x^0 = 10e$ # iterations (CPU Time in sec)	$x^0 = 16e$ # iterations (CPU Time in sec)	$x^0 = 30e$ # iterations (CPU Time in sec)
20.1	235(270.9)	223(288.2)	218(252.8)	217(231.7)
20.2	1007(1258.5)	1031(1317.8)	1094(1291.9)	573(749.7)
20.3	104(136.2)	99(128.3)	99(128.3)	148(193.1)
20.4	151(196.2)	108(139.9)	160(206.4)	193(249.3)
20.5	96(125.6)	126(163.6)	123(159.0)	106(139.4)

Table 4. Comparison with different initial solutions ($e = (1, \ldots, 1)$).

PROB	SOR-Newton		GSNA		*NBSOR*	
	#iterations	CPU Time(sec)	#iterations	CPU Time(sec)	#iterations	CPU Time(sec)
50.1	1493	9846.8	775	871.0	54	286.6
50.2	3953	25770.3	2739	2499.4	59	280.9
50.3	416	3076.1	3480	3224.8	49	242.1
50.4	273	1799.0	242	417.4	26	78.8
50.5	597	3938.5	673	920.3	34	116.6

Table 5. Comparative performance on mid-size problems.

PROB	# iterations	CPU Time (sec)
50.6	46	186.2
50.7	39	144.2
50.8	28	89.6
50.9	44	174.5
50.10	114	1014.1
50.11	66	339.7
50.12	18	50.4
50.13	61	297.0
50.14	49	211.2
50.15	31	102.2
50.16	22	65.9
50.17	39	153.9
50.18	35	143.1
50.19	41	178.9
50.20	120	1208.5

Table 6. Performance of $NBSOR$ on mid-size problems.

PROB	*NBSOR* # iterations	sec CPU
100.1	133	3640.1
100.2	64	992.6
100.3	51	644.5
200.1	30	886.9*

* $\epsilon = 10^{-2}$.

Table 7. Large-size Problems.

# iter.	PROB 50.21	PROB 50.22	PROB 50.23	PROB 50.23
5	16	18	17	17
10	.014	.014	.014	.018
25	2×10^{-3}	9×10^{-3}	4×10^{-4}	5×10^{-3}
29	2×10^{-3}	9×10^{-3}	8×10^{-6}	5×10^{-3}
45	2×10^{-3}	4×10^{-6}		4×10^{-5}
46	7×10^{-4}			7×10^{-6}
50	7×10^{-4}			
79	8×10^{-6}			

Table 8. Convergence of *NBSOR* to equilibrium.

PROB	*NBSOR* $\epsilon = 10^{-5}$ # iter.	GSNA $\epsilon = 10^{-5}$ # iter.	SOR-Newton $\epsilon = 10^{-1}$ # iter.
20.1	5	8	15
# active flows	400	400	252
CPU(sec)	9.1	15.1	64.1
20.2	5	8	15
# active flows	400	400	252
CPU(sec)	9.0	14.7	63.1
20.3	5	8	15
# active flows	400	400	252
CPU(sec)	9.0	14.7	62.9
20.4	5	8	15
# active flows	400	400	252
CPU(sec)	9.0	14.7	63.1
20.5	5	8	15
# active flows	400	400	252
CPU(sec)	9.0	14.6	63.2
30.1	6	10	91
# active flows	900	900	545
CPU(sec)	27.1	40.4	4555.2
40.1	6	12	
# active flows	1600	1600	
CPU(sec)	61.4	109.2	>1100min
50.1	7	12	
# active flows	2500	2500	
CPU(sec)	119.2	162.6	>600min

Table 9. Performance with nonconstant transportation costs